Bolt Beranek and Newman Inc.

Report No. 4368

LEVEL II

AD A084454

DTIC
ELECTE
MAY 9 1980
S D
C

BMG
Reference Manual

Norton Greenfeld, Frank Zdybel, Eugene Ciccarelli, and Martin Yonke

April 1980

Prepared for:
Defense Advanced Research Projects Agency

(14) BBN-

Report No. 4368

(12)

(6) BMG

Reference Manual

(9) Technical rept. 1 Jan 77 - 28 Mar 80

(10) Norton Greenfeld, Frank Zdybel, Eugene Ciccarelli, Martin Yonke

(11) April 1980 (12) 56

(15) This research has been supported by the Defense Advanced Research Projects Agency under Contract N00039-79-C-0316, ARPA Order No. 3740 and monitored by the Naval Electronics System Command (Program Code 9D30).

060 190

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A084 452 / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) BMG Reference Manual	5. TYPE OF REPORT & PERIOD COVERED Technical Report - 20 MAR 79 - 19 MAR 80
	6. PERFORMING ORG. REPORT NUMBER 4368
7. AUTHOR(s) N. Greenfeld, F. Zdybel, E. Ciccarelli, M. Yonke	8. CONTRACT OR GRANT NUMBER(s) N00039-79-C-0316
9. PERFORMING ORGANIZATION NAME AND ADDRESS Bolt Beranek and Newman Inc. 10 Moulton Street Cambridge, MA 02238	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS DARPA 1400 Wilson Blvd. Arlington, VA 22209	12. REPORT DATE April 1980
	13. NUMBER OF PAGES 65
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Naval Electronics System Command Program Code 9D30 Washington, D.C. 20360	15. SECURITY CLASS. (of this report) Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution of this report is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Distribution of this abstract is unlimited.

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Bit-Mapped Graphics, Computer Graphics, Raster-scan display, Interlisp

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This is the second design and implementation round of the BMG (for BitMapped Graphics) package in use at Bolt Beranek and Newman Inc's. Information Sciences Division for experimentation with the concepts and limitations of raster scan graphics systems. It is being used by research projects in command and control systems, natural language systems, and other advanced user interface work. This reference manual for the BMG system

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

is intended primarily for Interlisp programmers who wish to interface an application directly to the graphics system.

TABLE OF CONTENTS

PREFACE

This reference manual for the BBN BitMapped Graphics (BMG) system is intended primarily for Interlisp programmers who wish to interface an application directly to the graphics system. It serves the secondary purpose of an introductory manual for the non-Interlisp programmer who needs to build another interface to the graphics system using the low-level stream protocol between machines. Finally, it serves as an overview for those graphics-knowledgable systems designers who would like general information about the structure of this graphics system.

1. INTRODUCTION

This is the second design and implementation round of the BMG (for BitMapped Graphics) package in use at BBN's Information Sciences Division for experimentation with the concepts and limitations of raster scan graphics systems. The system is expressly designed for area-type manipulations, much in the spirit of [Teitelman, 1977] and [Sproull, 1979]. It is being used by research projects in command and control systems, natural language systems, and other advanced user interface work.

The BMG design presented here is the outgrowth of discussions among Eugene Ciccarelli, Norton Greenfeld, Martin Yonke, and Frank Zdybel. Frank defined the functional interface described below; Norton implemented the Interlisp functions; and Gene implemented the PDP-11 package which supports the system.

This design should be considered an intermediate one, since it is part of an evolving view of advanced raster graphics systems. It also represents a compromise between our conceptual desires and the reality of the hardware currently available.

This report documents the functional interface for user programs written in Interlisp as well as the communication protocol between the PDP-11 and the Decsystem-20. The latter is included in order that other interfaces can be written.[1]

Fully current and more detailed information can be found in the following files, all in directory <FONTWORK> on system BBN-TENEXD:

```
BMG.              Interlisp source code
BMG.COM           Compiled version of BMG with type checks
BMG.NODECL-COM    Compiled version of BMG without type checks
BMG.DESCRIBED     Documentation for BMG functions
BMG11.M11         Source code for BMG11
BMG11.INFO        Documentation for BMG11
BMG11-SPECS.DOC   Documentation for the BMG/BMG11 protocol
```

Hardware

The hardware currently available consists of a low-resolution (576x454) black and white bitmap system (with two independent planes and monitors) connected to a PDP-11/35, which in turn is connected via a low-speed line (9600 baud) to a Decsystem-20. The PDP-11 is used as a (somewhat large) graphics terminal and manages the bitmap itself, the keyboard, a digitizer tablet, and communication with the Decsystem-20. User programs run in that larger system.

[1] An interface to the circuit-design system SUDS has already been implemented.

The PDP-11/35 has 192K bytes of memory and an 80Mbyte disk (not used for anything other than bootstrapping). It also has a writable control storage (not used by the BMG system) and a BBN-built pager for 22-bit virtual addresses (used to reference memory above the first 64K bytes). The bitmap is produced by Symbolic Systems Inc., and allows the processor to reference the bitmap as either a device on the Unibus or as part of memory. Other I/O devices include a keyboard, a Summagraphics BITPAD tablet, and a serial line to a Decsystem-20.

In this paper, the term "BMG" refers to the Interlisp system residing in the Decsystem-20, while "BMG11" refers to the PDP-11 support system.

Overview

Conceptually, the system addresses planes and regions which are rectangular areas of bitmap. Planes can be considered to be "real" pieces of memory; regions are logical areas within a plane. The coordinate system sense is the normal one: the lower left corner is 0,0; the Y direction increases positively going up; the X direction increases positively going right.

There are two real-screen planes, 0 and 1. Both are the same size, with X ranging from 0 through 575,[2] and Y from 0 through 453. These values are inclusive -- points with X = 0 and X = 575 lie on the screen, similarly for Y = 0 and Y = 453. These planes are directly tied to video monitors (through the bitmap hardware), with a 0 or 1 bit value seen visually as either a white or black pixel (depending on current settings via BMGVideoSense).

There is one off-screen memory plane, 2. This is twice as high as the real-screen planes, though it has the same width. Thus for plane 2, X ranges from 0 through 575, Y from 0 through 907. This plane is not directly related to any video monitor, but can be used for local (PDP-11) storage of bitmaps.

Every region has properties relating to its location (its plane and limits), its graphics functions (its current internal position, display mode, current font, background pattern), and its general text functions (its internal scroll height, line spacing, left margin, and TTY cursor character). Regions may overlap each other.

All addressing is region-relative (except for defining the region limits). All graphic objects displayed in a region are clipped at the region's boundaries, except for "unescorted text" sent to the TTY region, which is "wrapped around" to the next line.

At all times there are two streams of output going from BMG to BMG11: graphics commands and general "unescorted" text. Consequently, there are two distinguished regions (which can be

[2] All numbers mentioned are in decimal, except in the Appendix.

changed dynamically), the "current graphics region" and the "current TTY region". All normal output to the terminal will go into the current TTY region; graphics output is done using the BMGxxx functions described below.

Similarly, there are two streams of input: normal type-in and "events" such as a timeout or a button hit on the digitizer. All four of these logical streams are multiplexed over a single TOPS20 terminal line[3].

[3] Since neither Interlisp nor TOPS20 particularly help with such multiplexing, there are circumstances of which a user must be careful. They will be described at the appropriate points below.

2. FUNCTIONALOVERVIEW

The following are the Interlisp functions which interface to the BMG system. All parameters and values are type-checked (using the DECL package [Teitelman, 1978]). The type declarations are contained in the next section. A non-type-checked BMG version is also available for application systems which are highly debugged and require maximum efficiency.

The following sections are organized by functional group; for easy reference, complete descriptions can be found in the alphabetical listing (pp.19-40).

2.1 Initializationand Synchronization

BMGInit [] Starts up BMG, resetting terminal type.

BMGReset [wipeFlg]
Resets state of system to default.

BMGResetTTY [regionNumber]
Clears the region and sets the current position at the appropriate point for text.

BMGShowPlane [planeNumber]
Returns a BMGPlaneSpecListRecord if the plane is defined, and NIL otherwise.

BMGInputP [] Checks to see whether there are any un-processed events on BMGEventList.

BMGOutFlush []
Flushes the Interlisp, TENEX and BMG output buffers.

BMGInFlush [] Discards the input events waiting on BMGEventList and resets it to NIL. Also clears the BMG read buffer (for general terminal input).

BMGEventFlush []
Discards the input events waiting on BMGEventList and resets it to NIL.

BMGVideoSense [planeNumber; videoSense]
Sets the video sense of the designated plane, returning the previous setting. If videoSense is null, the current setting is returned without side-effect.

BMGBellEnable [enableSetting]
Tells the system what action to take upon encountering a control-G in the TTY stream.

2.2 ManipulatingCharacteristics of Display Regions

Note that for the current implementation, all regions are forced to have a left boundary which is a multiple of 16 (i.e. the value given is truncated) and a right boundary which is one less than a multiple of 16 (increased to the next such number). The top and bottom boundaries can be any value. This restriction has been made for efficiency reasons, and will be removed in the next implementation.

BMGDefineRegion [regionSpecList]
: This function combines allocating a region number, locating the region, and specifying its characteristics.

BMGGetRegionNumber []
: Finds and returns an unassigned display region number.

BMGFreeRegionNumber[regionNumber]
: Declares that the region associated with the given region number is of no further interest by freeing the region number for re-assignment via BMGGetRegionNumber.

BMGLocateRegion [planeNumber; minX; maxX; minY; maxY; regionNumber]
: Locates the region on the designated plane with the given boundaries.

BMGDisplayMode [displayMode; regionNumber]
: Changes the display mode of the region.

BMGBackground [background; regionNumber]
: Sets the background shade of the region, returning the old value.

BMGFont [fontNumber; regionNumber]
: Sets the font of the region, returning the previous value.

BMGScrollHeight [scrollHeight; regionNumber]
: Sets the scrolling height for the region, returning the previous value.

BMGLineSpacing [lineSpacing; regionNumber]
: Sets the value of line feed (in terms of raster lines) for the region.

BMGLeftMargin [leftMargin; regionNumber]
: Sets the left margin for text for the region.

2.3 Interrogatingthe States of Display Regions

BMGShowRegion [regionNumber]

Generates a region specification list of the kind used by BMGDefineRegion, to describe the specified region.

BMGX [regionNumber]

Returns the x coordinate of the current position within the region (in region relative coordinates.)

BMGY [regionNumber]

Returns the y coordinate of the current position within the region (in region relative coordinates.)

BMGMinX [regionNumber]

Returns the x coordinate of the left hand edge of the region (in plane relative coordinates.)

BMGMaxX [regionNumber]

Returns the x coordinate of the right hand edge of the region (in plane relative coordinates.)

BMGMinY [regionNumber]

Returns the y coordinate of the lower edge of the region (in plane relative coordinates.)

BMGMaxY [regionNumber]

Returns the y coordinates of the upper edge of the region (in plane relative coordinates.)

2.4 ManipulatingDisplay Regions

BMGScrollRegion [scrollHeight; regionNumber]

Causes the region to scroll upwards by scrollHeight.

BMGCopyRegion [srcRegionNumber; destRegionNumber]

Fills the destination region with the contents of the source region.

BMGFillRegion [background; regionNumber]

Fills the region with the specified background.

2.5 GraphicOutput

BMGRegion [regionNumber]

Sets the current graphics region to regionNumber, returning the number of the previous graphics region.

BMGTTYRegion [regionNumber]

Sets the current TTY region to be regionNumber, returning the number of the previous TTY region.

BMGMove [xCoord; yCoord; regionNumber]

Makes (xCoord,yCoord) in region relative coordinates be the current position within the region.

BMGMoveRel [deltaX; deltaY; regionNumber]

Changes the current position within the region by the amount specified by deltaX and deltaY.

BMGPoint [xCoord; yCoord; regionNumber]

Draws a point (sensitive to the display mode) in the region at the location specified in region relative coordinates.

BMGPointRel [deltaX; deltaY; regionNumber]

Changes the current position by the amount specified by deltaX and deltaY and draws a point according to the preset display mode.

BMGLine [pos1XCoord; pos1YCoord; pos2XCoord; pos2YCoord; regionNumber]

Draws a line between the two specified positions in the region, according to the preset display mode.

BMGLineRel [deltaX; deltaY; regionNumber]

Draws a line from the current position in the region to the relative position specified by deltaX and deltaY according to the preset display mode.

BMGPolyLine [endPoints; regionNumbe]

Draws a connected, segmented line thru the endPoints starting at the current position.

BMGPolyLineRel [deltas; regionNumber]

Draws a connected, segmented line thru the positions starting at the current position.

BMGCircle [radius; regionNumber]

Draws a circle at the current location in the region.

BMGEllipse [semiMinorRadius; semiMajorRadius; orientation; regionNumber]

Draws an ellipse at the current location in the region.

BMGChar [charCode; regionNumber]
Draws the character designated by charCode from the current font set at the current position in the region.

BMGString [string; regionNumber]
Prints (using PRIN3) the given string with the current font starting at the current location in the region.

2.6 GraphicInput and Cursor Control

BMGTabletEnable [enableSetting]
Used to activate or deactivate the graphic tablet without disturbing the tablets' settings as defined via BMGTabletCursorState and BMGTabletButtonState.

BMGTabletCursorState[cursorState]
Sets the cursor state for the graphics cursor.

BMGTabletButtonState[enableMask]
Used to control the behavior of the graphic tablet.

BMGTabletRegion [regionNumber]
Sets the BMG region which will act as a confining boundary for the tablet cursor, as well as provide which plane the cursor will appear on.

BMGSetTimer [setting]
Starts the graphics process timer independently of the tablet timer.

BMGGetEvent [wait]
Returns a tablet, a graphics process or tablet or timer event, or NIL if none is waiting.

BMGSpliceEvent [eventSplice]
Arranges for splicing a string into the TTY input stream to announce the occurance of an event.

BMGReadTabletPosition []
Returns an event record describing the current state of the tablet.

BMGTTYCursorEnable[enableSetting]
Enables or disables the showing of the TTY cursor in the current TTY region.

BMGTTYCursorChar [charCodeRecord; regionNumber]

If charCodeRecord is non-NIL, sets the TTY cursor shape (as a character code) for use when region is the TTY region.

2.7 ManipulatingFonts

Currently available fonts are in directory <FONTS>. An implementation restriction limits the width of a character to 16 bits.

BMGDefineFont [fontNumber; fontDescriptor; sysResetFlg]
Downloads the specified font from the Decsystem20.

BMGDefineFontChar [char; fontNumber; charHeight; charWidth; charMapArray]
Specifies a character in a font.

BMGMakeFontFile [fontNumber; fontFileName]
Causes a Decsystem-20 file suitable for redefining the designated font to be written.

BMGDescribeFont [fontNumber]
Returns the loaded font descriptor corresponding to fontNumber (or NIL if none).

BMGGetFontNumber []
Returns an unused font number (i.e., a font number for which no BMGDefineFont has been performed since the last BMGInit, or for which a BMGFreeFontNumber has been performed.)

BMGFreeFontNumber [fontNumber]
In effect, declares that the font associated with the given font number is of no further interest by freeing the font number for re-assignment via BMGGetFontNumber.

2.8 ManipulatingDisplay Macros and Display Processes

A display macro is a sequence of graphics commands. A display process is an independent process within BMG11 repeatedly executing some display macro. One display macro may call another, though they are not recursive. (They don't have enough computational power to be -- e.g. no conditionals.) The process which executes graphics commands as they arrive from BMG is a special display process called the "graphics process". It is the only one that is initially created, and it cannot be killed. Display macros may be executed within the graphics process or within any display process. Each display process (and the graphics process) have their own "graphics region". A BMGRegion command changes the graphics region for the display process executing that command only.

BMGDefineMacro [macroNumber; startFlg]
Brackets a display macro definition.

BMGExpandMacro [macroNumber; iterationCount; iterationInterval]
Causes the macro definition associated by BMG with macroNumber to be expanded iterationCount times at a given frequency.

BMGKillMacros []
Kills all display macro definitions, and frees all display macro numbers.

BMGShowMacros []
Returns a list of the currently defined display macro numbers.

BMGGetMacroNumber []
Allocates an unused BMGMacroNumber if any are available, otherwise returns NIL.

BMGFreeMacroNumber [macroNumber]
In effect, declares that the macro associated with the given macro number is of no further interest by freeing the macro number for re-assignment via BMGGetMacroNumber.

BMGSpawnProcess [processNumber; macroNumber; iterationCount; iterationInterval; regionNumber]
Creates and starts a display process in the BMG11.

BMGPause [delayTime]
Pauses the current display process for the specified time.

BMGKillProcess [processNumber]
Kills the display process associated by BMG with processNumber.

BMGGetProcessNumber []
Allocates an unused ProcessNumber.

BMGFreeProcessNumber[processNumber]
In effect, declares that the process associated with the given process number is of no further interest by freeing the process number for re-assignment via BMGGetProcessNumber.

3. STRUCTUREDEFINITIONS

These are the formats of the (generally) internal records used by BMG. A user program should deal mainly with the "spec list" records.

3.1 RecordDeclarations

```
(PROPRECORD BMGPlaneSpecListRecord
    (PLANE# VIDEOSENSE MINX MINY MAXX MAXY OFFSCREENFLG))

(PROPRECORD BMGRegionSpecListRecord
    (REGION# PLANE MINX MAXX MINY MAXY CURRENTX CURRENTY
     DISPLAYMODE BACKGROUND SCROLLHEIGHT LINESPACING
     LEFTMARGIN FONT TTYCURSORCHAR))

(TYPERECORD BMGPlaneRecord
    (Plane\PlaneNumber Plane\PlaneVideoSense Plane\MinX Plane\MaxX
     Plane\MinY Plane\MaxY Plane\OffScreenFlg)
           Plane\OffScreenFlg _T)

(TYPERECORD BMGRegionRecord
    (Region\Number Region\PlaneNumber Region\MinX Region\MaxX
     Region\MinY Region\MaxY Region\CurrentX Region\CurrentY
     Region\DisplayMode Region\Font Region\Background
     Region\ScrollHeight Region\LineSpacing Region\LeftMargin
     Region\TTYCursorChar))

(TYPERECORD BMGLoadedFontDescriptorRecord
    (LoadedFont\Number LoadedFont\FileName LoadedFont\FontRecord))

(TYPERECORD BMGFontRecord
    (Font\Name Font\Height Font\BaseLine
     Font\ColumnPositionAdjustment Font\CharArray))

(TYPERECORD BMGCharCode
    (CharCode\Font CharCode\Code))

(TYPERECORD BMGCharDefRecord
    (CharDef\Code CharDef\RasterWidth CharDef\Width
     CharDef\LeftKern CharDef\RasterArray))
```

```
(TYPERECORD BMGProcessRecord
    (Process\Number Process\Macro Process\Region
     Process\IterationCount Process\IterationInterval))

(RECORD BMGEventRecord
     (Event\Name . Event\Parameters)
         the following types can occur:
              (DownTransition Tablet\X Tablet\Y)
              (UpTransition   Tablet\X Tablet\Y)
              (TabletTimeout  buttonState Tablet\X Tablet\Y)
              (GraphicsTimeout)

(TYPERECORD BMGTabletCursorStateRecord
     (Cursor\UpChar Cursor\DownChar))
            note: both UpChar and DownChar are BMGCharCode records.

(TYPERECORD BMGButtonEnableRecord
    (Button\UpMask Button\DownMask Button\TimerMask Button\Timeout)
            Button\UpMask _0 Button\DownMask _0
            Button\TimerMask _0 Button\Timeout _0)
```

3.2 DECLtypes

These type declarations are defined by and enforced by the DECL package (see Section 24.20 in [Teitelman, 1978]). The BMG.NODECL-COM file has been compiled with all type-checking deleted, but should only be used after extensive debugging of an application program.

```
BMGRegionNumber    [SMALLP (SATISFIES
                            (AND (ILEQ VALUE var(BMGMaxRegionNumber))
                                 (NOT (FMEMB VALUE BMGFreeRegionNumList]

BMGPlaneNumber     [SMALLP (SATISFIES
                            (AND (ILEQ VALUE BMGMaxPlaneNumber)
                                 (NOT (FMEMB VALUE BMGFreePlaneNumList]

BMGScreenCoord     [FIXP (SATISFIES (AND (ILEQ VALUE 32767)
                                         (IGEQ VALUE -32768]

CharCode           [SMALLP (SATISFIES (AND (IGEQ VALUE 0)
                                           (ILEQ VALUE 128]
                                     (* Note: the fonts we currently have
                                        (on the PDP11) have 129! characters)

BMGMacroNumber     [SMALLP (SATISFIES
                           (AND (ILEQ VALUE BMGMaxMacroNumber)
                                (NOT (FMEMB VALUE BMGFreeMacroNumList]
```

```
BMGProcessNumber  [SMALLP (SATISFIES
                           (AND (ILEQ VALUE BMGMaxProcessNumber)
                                (NOT (FMEMB VALUE BMGFreeProcessNumList]

BMGFontNumber     [SMALLP [SATISFIES
                          (AND (IGEQ VALUE 0)
                               (ILEQ VALUE BMGMaxFontNumber)
                               (NOT (FMEMB VALUE BMGFreeFontNumList]


BMGDisplayMode    [MEMQ ADD REMOVE FLIP OVERWRITE INVERTEDOVERWRITE]

BMGVideoSense     [MEMQ WHITEONBLACK BLACKONWHITE]
```

4. FUNCTIONDESCRIPTIONS

The following are the full descriptions of all BMG functions, cross-referenced to the function-grouping list above.

"BMGBackground" [background; regionNumber] {p. 8}

description:

Sets the background shade of the region, returning the old value. Only the low order 16 bits are used to specify a four pixel square region. The background shade is used when filling the region, in scrolling, and for erasing characters. If background is null, the current background value is returned without side effect. Uses the current graphics region if regionNumber is NIL.

parameters:

background	either a FIXP, or NIL
regionNumber	either a BMGRegionNumber, or NIL.

returns: a FIXP.

"BMGBellEnable" [enableSetting] {p. 7}

description:

Tells the system what action (if any) to take upon encountering a control-G in the TTY stream. Always returns the previous setting, and changes it only if enableSetting is non-NIL.

parameters:

enableSetting	either NIL, 'OFF, 'AUDIO, or 'VISUAL.

returns: either 'OFF, 'AUDIO, or 'VISUAL.

"BMGChar" [charCode; regionNumber] {p. 11}

description:

Draws the character designated by charCode from the current font set at the current position in the region. Does not disturb Interlisp's line position counter. Does not affect the current position in the region. Uses the current graphics region if regionNumber is NIL.

parameters:

charCode	a CharCode.
regionNumber	either a BMGRegionNumber, or NIL.

returns: NIL.

"BMGCircle" [radius; regionNumber] {p. 10}

description:

Draw a circle at the current location in the region. The current location in the region is left at the center of the circle. Uses the current graphics region if regionNumber is NIL.

parameters:

radius	a FIXP that satisfies (IGEQ radius 0)
regionNumber	either a BMGRegionNumber, or NIL.

returns: NIL.

"BMGCopyRegion" [srcRegionNumber; destRegionNumber] {p. 9}

description:

If destRegionNumber is null, the current graphics region is assumed to be the destination. The destination region is filled with the contents of the source region. The copying process is sensitive to the display mode of the destination region. The regions do not have to be the same size - the regions' origins are aligned and the destination region determines the size of the copy.

parameters:

srcRegionNumber	a BMGRegionNumber.
destRegionNumber	either a BMGRegionNumber, or NIL.

returns: NIL.

"BMGDefineFont" [fontNumber; fontDescriptor; sysResetFlg] {p. 12}

description:

Downloads the specified font from the Decsystem20. If fontDescriptor is a LITATOM, it is taken as a file name and that file is loaded. If it is a BMGFontRecord, a blank font is established for later manipulation via BMGDefineFontChar. The value returned is the previous descriptor for the given font number, unless we were unsuccessful at downloading the new font. In this case, it is the PDP11 response reason. This response is a list: the first element is 'ERROR and if we couldn't load the font the second element is 'FONT while the third is the reason given. If we couldn't load a character, the second element is the char number, while the third is the reason.

parameters:

fontNumber	a FIXP that satisfies (AND (IGEQ fontNumber 0) (ILEQ fontNumber BMGMaxNumberOfFonts))
fontDescriptor	either a BMGFontRecord, or a LITATOM.
sysResetFlg	either NIL, or T. *[for use only by BMG system fns]*

returns: either a BMGLoadedFontDescriptorRecord, or a non-NIL list that satisfies
(VALUE:1 = 'ERROR)

"BMGDefineFontChar" [char; fontNumber; charHeight; charWidth; charMapArray] {p. 12}

description:

Used for specifying a character in a font. charWidth and charHeight give the true dimensions of the font cell. In order to prevent overflow of the space actually available on the BMG11 for storing the character, the program must previously have assured sufficient room in the font for the character. Similarly, the amount of room for the character in BMG11 may be in excess of the actual size of the character cell. However, this situation can be corrected by making a new font file and downloading it. In any case, BMG's descriptor for the font is immediately adjusted to reflect the true size of charMapArray.

parameters:

char	a CharCode.
fontNumber	a BMGFontNumber.
charHeight	a SMALLP that satisfies (IGEQ charHeight 0)
charWidth	a SMALLP that satisfies (IGEQ charWidth 0)
charMapArray	an ARRAYP.

returns: a BMGCharCode that satisfies
(AND VALUE:CharCode\Font = fontNumber
VALUE:CharCode\Code = char)

"BMGDefineMacro" [macroNumber; startFlg] {p. 13}

description:

Brackets a display macro definition. If startFlg is START, then this function will either use the macro number given or allocate a new macro number. It will delete the definition, if it exists, of the current display macro with that number. After this call, all display primitives called will be collected as part of the definition of this macro, until this function is called with startFlg END. On either call, a successful completion is indicated by a return value of the macro number being defined, and an error indication by a list of 'ERROR and the reason.

parameters:

macroNumber either a BMGMacroNumber, or NIL.
startFlg either 'START, or 'END.

returns: either a BMGMacroNumber, or a non-NIL list that satisfies (VALUE:1 = 'ERROR)

"BMGDefineRegion" [regionSpecList] {p. 8}

description:

This function combines allocating a region number, locating the region, and specifying its characteristics. If an oldRegionNumber (i.e. REGION# in the regionSpecList) is given and that region is currently defined, the specifications in regionSpecList are applied to change the region's characteristics (e.g., FONT, BACKGROUND, DISPLAYMODE.) If oldRegionNumber is not given, is NIL, or is not a currently defined region, the specifications must include the plane on which it is to reside. The number of the region is returned as the value. Note that the defaults for region parameters are generally specified by some of the variables in BMGDEFAULTVARS:
(BMGDefaultRegionFontNumber BMGDefaultRegionBackground
BMGDefaultRegionDisplayMode BMGDefaultRegionLineSpacing
BMGDefaultRegionLeftMargin BMGDefaultRegionTTYCursorChar)
though the default ScrollHeight is 1/2 the region total height, and the boundaries default to the plane's boundaries.

parameters:

regionSpecList a BMGRegionSpecListRecord.

returns: either a BMGRegionNumber, or NIL.

"BMGDescribeFont" [fontNumber] {p. 12}

description:

Returns the loaded font descriptor corresponding to fontNumber (or NIL if none).

parameters:

fontNumber	a FIXP that satisfies (AND (IGEQ fontNumber 0) (ILEQ fontNumber BMGMaxNumberOfFonts))

returns: either a BMGLoadedFontDescriptorRecord, or NIL.

"BMGDisplayMode" [displayMode; regionNumber] {p. 8}

description:

Changes the display mode of the region if display mode is given, returning the previous value. If displayMode is null, returns the current setting without side effect. Uses the current graphics region if regionNumber is NIL.

parameters:

displayMode	either a BMGDisplayMode, or NIL.
regionNumber	either a BMGRegionNumber, or NIL.

returns: a BMGDisplayMode.

"BMGEllipse" [semiMinorRadius; semiMajorRadius; orientation; regionNumber] {p. 11}

description:

Draws an ellipse at the current location in the region. At orientation 0 the semimajor axis is horizontal, the semiminor axis vertical. The current location in the region is left at the center of the ellipse. Uses the current graphics region if regionNumber is NIL. (Note: currently computed by BMG, rather than BMG11.)

parameters:

semiMinorRadius	a FIXP that satisfies (IGREATERP semiMinorRadius 0)
semiMajorRadius	a FIXP that satisfies (IGREATERP semiMajorRadius 0)
orientation	a FIXP that satisfies (AND (IGEQ orientation -90) (ILESSP orientation 90))
regionNumber	either a BMGRegionNumber, or NIL.

returns: NIL.

"BMGEventFlush" [] {p. 7}

description:

Discards the input events waiting on BMGEventList and resets it to NIL.

returns: NIL.

"BMGExpandMacro" [macroNumber; iterationCount; iterationInterval] {p. 13}

description:

Causes the macro definition associated by BMG with macroNumber to be expanded iterationCount times at a given frequency. iterationCount = 0 means expand indefinitely. iterationInterval is in 100ths of a second, and is measured from the start of one expansion of the macro to the start of the next.

parameters:

macroNumber	a BMGMacroNumber.
iterationCount	a SMALLP that satisfies (AND (IGEQ iterationCount 0) (ILEQ iterationCount 255))
iterationInterval	a FIXP.

returns: a BMGMacroNumber that satisfies
(VALUE = macroNumber)

"BMGFillRegion" [background; regionNumber] {p. 9}

description:

Fills the region with the specified background. If background is null, the background shade specified for the region via BMGBackground will be used. Uses the current graphics region if regionNumber is NIL.

parameters:

background	either a FIXP, or NIL.
regionNumber	either a BMGRegionNumber, or NIL.

returns: NIL.

"BMGFont" [fontNumber; regionNumber] {p. 8}

description:

Sets the font of the region, returning the previous value. If fontNumber is null, the current font number is returned without side effect. Uses the current graphics region if regionNumber is NIL.

parameters:

fontNumber either a BMGFontNumber, or NIL.
regionNumber either a BMGRegionNumber, or NIL.

returns: a BMGFontNumber.

"BMGFreeFontNumber" [fontNumber] {p. 12}

description:

In effect declares that the font associated with the given font number is of no further interest by freeing the font number for re-assignment via BMGGetFontNumber.

parameters:

fontNumber a BMGFontNumber.

returns: NIL.

"BMGFreeMacroNumber" [macroNumber] {p. 13}

description:

In effect declares that the macro associated with the given macro number is of no further interest by freeing the macro number for re-assignment via BMGGetMacroNumber. The macro number nevertheless remains defined at the BMG11 as a string of BMG11 command escapes, thus taking up PDP11 memory space (an implementation deficiency to be remedied at a later date). Note that BMGKillMacros reclaims the space allocated to them all.

parameters:

macroNumber a BMGMacroNumber.

returns: NIL.

"BMGFreeProcessNumber" [processNumber] {p. 13}

description:

In effect declares that the process associated with the given process number is of no further interest by freeing the process number for re-assignment via BMGGetProcessNumber. This routine does not kill the process (see BMGKillProcess) and it may be still active in the PDP-11.

parameters:

processNumber a BMGProcessNumber.

returns: NIL.

"BMGFreeRegionNumber" [regionNumber] {p. 8}

description:

Declares that the region associated with the given region number is of no further interest by freeing the region number for re-assignment via BMGGetRegionNumber.

parameters:

regionNumber a BMGRegionNumber.

returns: NIL.

"BMGGetEvent" [wait] {p. 11}

description:

Returns a tablet, a graphics process or tablet or timer event, or NIL if none is waiting, except that if wait is T the function will hang until an event is generated.

parameters:

wait either NIL, or T.

returns: either NIL or a BMGEventRecord.

"BMGGetFontNumber" [] {p. 12}

description:

Returns an unused font number (i.e., a font number for which no BMGDefineFont has been performed since the last BMGInit, or for which a BMGFreeFontNumber has been performed.) Takes elements from BMGFreeFontNumList if that global is non-nul:, otherwise assigns the current value of BMGNumberOfFonts and increments its value and the value of BMGMaxFontNumber. If BMGNumberOfFonts is EQ to BMGMaxNumberOfFonts, returns NIL.

returns: either a BMGFontNumber, or NIL.

"BMGGetMacroNumber" [] {p. 13}

description:

Allocates an unused BMGMacroNumber if any are available, otherwise returns NIL. The number may still have a valid definition associated with it as far as BMG11 is concerned. Takes elements from BMGFreeMacroNumList if that global is non-null. Otherwise allocates BMGNumberOfMacros and increments the value of that global and BMGMaxMacroNumber. If BMGNumberOfMacros is EQ to BMGMaxNumberOfMacros, returns NIL.

returns: either a BMGMacroNumber, or NIL.

"BMGGetProcessNumber" [] {p. 13}

description:

Allocates an unused ProcessNumber. The number may still have a active process associated with it as far as BMG11 is concerned. Takes elements from BMGFreeProcessNumList if that global is non-null, otherwise allocates BMGNumberOfProcesses and increments the value of that global and of BMGMaxProcessNumber. If BMGNumberOfProcesses is EQ to BMGMaxNumberOfProcesses, returns NIL.

returns: either a BMGProcessNumber, or NIL.

"BMGGetRegionNumber" [] {p. 8}

description:

Finds and returns an unassigned display region number. Takes elements from BMGFreeRegionNumList if that global is non-null, otherwise assigns the current value of BMGNumberOfRegions and increments its value and the value of BMGMaxRegionNumber. If BMGNumberOfRegions is EQ to BMGMaxNumberOfRegions, returns NIL.

returns: either a BMGRegionNumber, or NIL.

"BMGInFlush" [] {p. 7}

description:

Discards the input events waiting on BMGEventList and resets it to NIL. Also clears the BMG read buffer (for general terminal input).

returns: NIL.

"BMGInit" [] {p. 7}

description:

If the global BMGTerminalType is NIL, first asks the user whether he is using the BBN Bit Map terminal and sets BMGTerminalType appropriately to either 'OTHER or 'BMG If necessary, sends appropriate commands to TOPS20 to set up properly for the BMG terminal. Calls BMGReset to do all other initialization.

"BMGKillMacros" [] {p. 13}

description:

Kills all display macro definitions, and frees all display macro numbers.

returns: NIL.

"BMGKillProcess" [processNumber] {p. 13}

description:

Kills the display process associated by BMG with processNumber. Frees the process number.

parameters:

processNumber a BMGProcessNumber.

returns: NIL.

"BMGLeftMargin" [leftMargin; regionNumber] {p. 8}

description:

Sets the left margin for the region, that is, the x position resulting after a carriage-return character is sent to this region as the current TTY region. The value given is relative to the left edge of the region. Returns the old value. If leftMargin is null, the current setting is returned witout side effect. Uses the current TTY region if regionNumber is NIL.

parameters:

leftMargin either a FIXP that satisfies
(IGEQ leftMargin BMGMinLeftMargin)
or NIL.

regionNumber either a BMGRegionNumber, or NIL.

returns: a FIXP.

"BMGLine" [pos1XCoord; pos1YCoord; pos2XCoord; pos2YCoord; regionNumber] {p. 10}

description:

Draws a line between the two specified positions in the region, according to the preset display mode. If pos2XCoord and pos2YCoord are NIL, draws a line from the current position to pos1XCoord, pos1YCoord. The current position becomes the second endpoint of the line. Uses the current graphics region if regionNumber is NIL.

parameters:

pos1XCoord	a BMGScreenCoord.
pos1YCoord	a BMGScreenCoord.
pos2XCoord	either a BMGScreenCoord, or NIL.
pos2YCoord	either a BMGScreenCoord, or NIL.
regionNumber	either a BMGRegionNumber, or NIL.

returns: NIL.

"BMGLineRel" [deltaX; deltaY; regionNumber] {p. 10}

description:

Draws a line from the current position in the region to the relative position specified by deltaX and deltaY according to the preset display mode. Uses the current graphics region if regionNumber is NIL.

parameters:

deltaX	a FIXP.
deltaY	a FIXP.
regionNumber	either a BMGRegionNumber, or NIL.

returns: NIL.

"BMGLineSpacing" [lineSpacing; regionNumber] {p. 8}

description:

Sets the value of line feed (in terms of screen coordinates) for the region, except that lineSpacing cannot be reduced below the sum of the maximum height of the current font and BMGMinInterLineSpacing. If lineSpacing is null, returns the current value without side effect. Uses the current TTY region if regionNumber is NIL.

parameters:

lineSpacing	either a FIXP, or NIL.
regionNumber	either a BMGRegionNumber, or NIL.

returns: a FIXP.

"BMGLocateRegion" [planeNumber; minX; maxX; minY; maxY; regionNumber] {p. 8}

description:

Locates the region on the designated plane with the given boundaries. Uses the current graphics region if regionNumber is NIL. Uses the region's current values for any parameters specified as NIL. Note that the current position after the locate is 0,0 (i.e. the lower left corner of the region). Returns the region number.

parameters:

planeNumber	either a BMGPlaneNumber, or NIL.
minX	either a BMGScreenCoord, or NIL.
maxX	either a BMGScreenCoord, or NIL.
minY	either a BMGScreenCoord, or NIL.
maxY	either a BMGScreenCoord, or NIL.
regionNumber	either a BMGRegionNumber, or NIL.

returns: a BMGRegionNumber.

"BMGMakeFontFile" [fontNumber; fontFileName] {p. 12}

description:

Causes a PDP-10 file suitable for redefining the designated font to be written. The FontDescriptor for the designated font is returned with updated font file name field. If the write is unsuccessful for any reason, returns NIL.

parameters:

fontNumber	a BMGFontNumber.
fontFileName	a LITATOM.

returns: either a BMGLoadedFontDescriptorRecord, or NIL.

"BMGMaxX" [regionNumber] {p. 9}

description:

Returns the x coordinate of the right hand edge of the region in plane relative coordinates. Uses the current graphics region if regionNumber is NIL.

parameters:

regionNumber	either a BMGRegionNumber, or NIL.

returns: a FIXP.

"BMGMaxY" [regionNumber] {p. 9}

description:

Returns the y coordinates of the upper edge of the region in plane relative coordinates. Uses the current graphics region if regionNumber is NIL.

parameters:

regionNumber	either a BMGRegionNumber, or NIL.

returns: a FIXP.

"BMGMinX" [regionNumber] {p. 9}

description:

Returns the x coordinate of the left hand edge of the region in plane relative coordinates. Uses the current graphics region if regionNumber is NIL.

parameters:

regionNumber either a BMGRegionNumber, or NIL.

returns: a FIXP.

"BMGMinY" [regionNumber] {p. 9}

description:

Returns the y coordinate of the lower edge of the region in plane relative coordinates. Uses the current graphics region if regionNumber is NIL.

parameters:

regionNumber either a BMGRegionNumber, or NIL.

returns: a FIXP.

"BMGMove" [xCoord; yCoord; regionNumber] {p. 10}

description:

Makes (xCoord, yCoord) in region relative coordinates be the current position within the region. Uses the current graphics region if regionNumber is NIL.

parameters:

xCoord a BMGScreenCoord.

yCoord a BMGScreenCoord.

regionNumber either a BMGRegionNumber, or NIL.

returns: NIL.

"BMGMoveRel" [deltaX; deltaY; regionNumber] {p. 10}

description:

Changes the current position within the region by the amount specified by deltaX and deltaY. Uses the current graphics region if regionNumber is NIL.

parameters:

deltaX a FIXP.

deltaY a FIXP.

regionNumber either a BMGRegionNumber, or NIL.

returns: NIL.

"BMGOutFlush" [] {p. 7}

description:

Flushes the Interlisp, TENEX and BMG output buffers.

returns: NIL.

"BMGPause" [delayTime] {p. 13}

description:

Pause the current display process for the specified time. delayTime given in 100ths of a second, and is 15 bits only (thus, the maximum delay time is around 6.5 seconds).

parameters:

delayTime	a FIXP.

returns: NIL.

"BMGPoint" [xCoord; yCoord; regionNumber] {p. 10}

description:

Draws a point (sensitive to the display mode) in the region at the location specified in region relative coordinates. Changes the current position to the given position. Uses the current graphics region if regionNumber is NIL.

parameters:

xCoord	a BMGScreenCoord.
yCoord	a BMGScreenCoord.
regionNumber	either a BMGRegionNumber, or NIL.

returns: NIL.

"BMGPointRel" [deltaX; deltaY; regionNumber] {p. 10}

description:

Changes the current position by the amount specified by deltaX and deltaY and draws a point according to the preset display mode. Uses the current graphics region if regionNumber is NIL.

parameters:

deltaX	a FIXP.
deltaY	a FIXP.
regionNumber	either a BMGRegionNumber, or NIL.

returns: NIL.

"BMGPolyLine" [endPoints; regionNumber] {p. 10}

description:

Draws a connected, segmented line thru the endPoints starting at the current position. endPoints can be either a list of (X . Y) cons cells, or a generator handle which will successively produce the (X . Y) pairs. The current position is left at the last end point in the list. Uses the current graphics region if regionNumber is NIL.

parameters:

endPoints	either a list of (CONS X Y) end point positions, or a generator handle. i.e. a non-NIL list that satisfies [OR [for entry in endPoints always (AND (type? BMGScreenCoord (CAR entry)) (type? BMGScreenCoord (CDR entry] [AND (STACKP (CAR endPoints)) (STACKP (CDR endPoints]]
regionNumber	either a BMGRegionNumber, or NIL.

returns: NIL.

"BMGPolyLineRel" [deltas; regionNumber] {p. 10}

description:

Draws a connected, segmented line thru the positions starting at the current position. deltas can be either a list of (deltaX,Y) cons cells, or a generator handle which will successively produce the (deltaX,Y) pairs. The current position is left at the last point drawn. Uses the current graphics region if regionNumber is NIL.

parameters:

deltas	either a list of (CONS deltaX deltaY) relative positions, or a generator handle. i.e. a non-NIL list that satisfies [OR [for entry in deltas always (AND (type? FIXP (CAR entry)) (type? FIXP (CDR entry] [AND (STACKP (CAR deltas)) (STACKP (CDR deltas]]
regionNumber	either a BMGRegionNumber, or NIL.

returns: NIL.

"BMGReadTabletPosition" [] {p. 11}

description:

Returns an event record describing the current state of the tablet, or NIL if the tablet is not turned on. If an error response is received from the BMG11, returns (LIST 'ERROR reason).

returns: either NIL or a BMGEventRecord.

"BMGRegion" [regionNumber] {p. 10}

description:

Sets the current graphics region to regionNumber, returning the number of the previous graphics region. If regionNumber is null, returns the number of the current grahics region without side effect.

parameters:

regionNumber either a BMGRegionNumber, or NIL.

returns: a BMGRegionNumber.

"BMGReset" [wipeFlg] {p. 7}

description:

Resets the planes to BMGDefaultPlaneRecords. If wipeFlg is non null, each plane is filled according to the value of the global variable BMGDefaultPlaneBackground. Then the initial state of the system is set according to the variables on BMGDEFAULTVARS. The ones which are used here are:
(BMGDefaultRegionSpecLists BMGDefaultCurrentRegion BMGDefaultCurrentTTYRegion BMGDefaultBellEnable BMGDefaultTabletCursorState BMGDefaultTabletRegion BMGDefaultTabletEnable BMGDefaultTabletButtonEnableMask BMGDefaultTTYCursorEnable BMGDefaultFont BMGDefaultSpliceEvent).

parameters:

wipeFlg either NIL, or T.

"BMGResetTTY" [regionNumber] {p. 7}

description:

This function clears the region and sets the current position within that region at the appropriate point for text (upper left, minus one baseline and to the right of the left margin by LeftMargin). Uses the current TTY region if regionNumber is NIL.

parameters:

regionNumber either a BMGRegionNumber, or NIL.

returns: NIL.

"BMGScrollHeight" [scrollHeight; regionNumber] {p. 8}

description:

Sets the scrolling height for the region, returning the previous value. This is the amount (in terms of screen coordinates) that the display region is scrolled upward when BMGScrollRegion is called (except that a scrollHeight of 0 means to clear the entire region and start at the top). Scrolling will also occur in the TTY region when text reaches the bottom of that region. In this case, a negative scroll height means "wait for the user to type control-RQ" and then scroll; a positive scroll height of zero means an automatic clear and start at the top of the region. If scrollHeight is null, the current value is returned without side effect. Uses the current TTY region if regionNumber is NIL.

parameters:

scrollHeight	either a FIXP, or NIL.
regionNumber	either a BMGRegionNumber, or NIL.

returns: a FIXP.

"BMGScrollRegion" [scrollHeight; regionNumber] {p. 9}

description:

Causes the region to scroll upwards by scrollHeight pixels if scrollHeight is non-null, otherwise by the amount set for the region via BMGScrollHeight. Uses the current TTY region if regionNumber is NIL.

parameters:

scrollHeight	either a FIXP, or NIL.
regionNumber	either a BMGRegionNumber, or NIL.

returns: NIL.

"BMGSetTimer" [setting] {p. 11}

description:

Starts the graphics process timer independently of the tablet timer. The timer setting is given in 100ths of a second. When the timer expires, an event is generated.

parameters:

setting	a FIXP that satisfies (IGREATERP setting 0)

returns: NIL.

"BMGShowMacros" [] {p. 13}

description:

returns a list of the currently defined display macro numbers.

returns: a list, or NIL.

"BMGShowPlane" [planeNumber] {p. 7}

description:

Returns a BMGPlaneSpecListRecord if the plane is defined, and NIL otherwise.

parameters:

planeNumber a FIXP.

returns: either a BMGPlaneSpecListRecord, or NIL.

"BMGShowRegion" [regionNumber] {p. 9}

description:

Generates a region specification list of the kind used by BMGDefineRegion, to describe the specified region. If regionNumber is null, the current graphics region is described. If there is no region of the given number, returns NIL.

parameters:

regionNumber either a FIXP, or NIL.

returns: a BMGRegionSpecListRecord, or NIL.

"BMGSpawnProcess" [processNumber; macroNumber; iterationCount; iterationInterval; regionNumber] {p. 13}

description:

Create and start a display process in the PDP-11, telling it to execute a given display macro n times at a given frequency (as if the display process had a top level BMGExpandMacro command) and use a given graphics region initially (the current graphics region if regionNumber is NIL). When the display process is done, it will kill itself. If the specified iterationCount is 0, the display process will execute the display macro indefinitely (until killed by a BMGReset, or killed specifically by a BMGKillProcess, etc.). The iterationInterval is in 100ths of a second (and is limited to 15 bits).

parameters:

processNumber	either a BMGProcessNumber, or NIL.
macroNumber	a BMGMacroNumber.
iterationCount	a SMALLP that satisfies (AND (IGEQ iterationCount 0) (ILESSP iterationCount 256))
iterationInterval	a FIXP.
regionNumber	either a BMGRegionNumber, or NIL.

returns: either a BMGProcessNumber, or a non-NIL list that satisfies (VALUE:1 = 'ERROR)

"BMGSpliceEvent" [eventSplice] {p. 11}

description:

Arranges for splicing a string into the TTY input stream to announce the occurance of an event. If eventSplice is a string, that string will be spliced into the input stream and the event record will still be available in the event queue. If eventSplice is EQ to 'RECORD the string spliced in will be the pname of the event record, and the event records will no longer be available on BMGEventList. If eventSplice is EQ to 'OFF the splice feature is turned off. If eventSplice is null, the previous value of eventSplice is returned without side effect.

parameters:

eventSplice	either a STRINGP, 'RECORD, or 'OFF, or NIL.

returns: either a STRINGP, or 'RECORD, or 'OFF.

"BMGString" [string; regionNumber] {p. 11}

description:

Prints (using PRIN3) the given string with the current font starting at the current location in the region. Does not affect Interlisp's current line position counter. Does not affect the current position in the region. Uses the current graphics region if regionNumber is NIL. The string is clipped (by whole characters) to stay in the region.

parameters:

string	anything.
regionNumber	either a BMGRegionNumber, or NIL.

returns: NIL.

"BMGTabletButtonState" [enableMask] {p. 11}

description:

Used to control the behavior of the graphic tablet. enableMask contains the enable bits for all of the 2-state devices on the system (currently only one button, for the Summagraphics BITPAD tablet.) The Up and Down masks give enables for event reporting for the respective up and down transitions of each device, and the timer mask gives the enables for starting the tablet timer on the transition. The old setting is returned.

parameters:

enableMask either a BMGButtonEnableRecord, or NIL.

returns: a BMGButtonEnableRecord.

"BMGTabletCursorState" [cursorState] {p. 11}

description:

Sets the cursor state for the graphics cursor, if cursorState is non-NIL. Returns the current state.

parameters:

cursorState either a BMGTabletCursorStateRecord, or NIL.

returns: a BMGTabletCursorStateRecord.

"BMGTabletEnable" [enableSetting] {p. 11}

description:

Used to activate or deactivate the graphic tablet without disturbing the tablets' settings as defined via BMGTabletButtonState and BMGTabletCursorState. When the tablet is off, the tablet cursor is not displayed and BMGReadTabletPosition returns NIL. Returns the previous activation state. If enableSetting is null, returns the old state without changing it.

parameters:

enableSetting either 'ON, 'OFF, or NIL.

returns: either 'ON, or 'OFF.

"BMGTabletRegion" [regionNumber] {p. 11}

description:

Sets the BMG region which will act as a confining boundary for the tablet cursor, as well as provide which plane the cursor will appear on. Returns the old region, and does not change it if the input is NIL.

parameters:
regionNumber either a BMGRegionNumber, or NIL.
returns: a BMGRegionNumber.

"BMGTTYCursorChar" [charCodeRecord; regionNumber] {p. 12}
description:
If charCodeRecord is non-NIL, sets the TTY cursor shape (as a character code) for use when region is the TTY region. Uses the current TTY region if regionNumber = NIL. Always returns the previous TTY cursor char for the region.
parameters:
charCodeRecord either a BMGCharCode, or NIL.
regionNumber either a BMGRegionNumber, or NIL.
returns: a BMGCharCode.

"BMGTTYCursorEnable" [enableSetting] {p. 11}
description:
Enables or disables the showing of the TTY cursor in the current TTY region. Returns the previous setting. If enableSetting is null, returns the current setting without changing it.
parameters:
enableSetting either NIL, 'ON, or 'OFF.
returns: either 'ON, or 'OFF.

"BMGTTYRegion" [regionNumber] {p. 10}
description:
Sets the current TTY region to be regionNumber, returning the number of the previous TTY region. If regionNumber is null, returns the number of the current TTY region without side effect.
parameters:
regionNumber either a BMGRegionNumber, or NIL.
returns: a BMGRegionNumber.

"BMGVideoSense" [planeNumber; videoSense] {p. 7}
description:
Sets the video sense of the designated plane, returning the previous setting. If videoSense is null, the current setting is returned without side-effect.

parameters:

planeNumber	a BMGPlaneNumber.
videoSense	either a BMGVideoSense, or NIL.

returns: a BMGVideoSense.

"BMGX" [regionNumber] {p. 9}

description:

Returns the x coordinate of the current position within the region (in region relative coordinates.) Uses the current graphics region if regionNumber is NIL.

parameters:

regionNumber	either a BMGRegionNumber, or NIL.

returns: a FIXP.

"BMGY" [regionNumber] {p. 9}

description:

Returns the y coordinate of the current position within the region (in region relative coordinates.) Uses the current graphics region if regionNumber is NIL.

parameters:

regionNumber	either a BMGRegionNumber, or NIL.

returns: a FIXP.

APPENDIX A
BMG COMMANDS ACCEPTED BY BMG11

This appendix details the protocol between BMG and BMG11. It is here for use in implementing non-BMG interfaces to BMG11. Note, numbers in this appendix are in octal unless otherwise indicated.

A.1 BMG11 Functions

```
- This column has the name of the BMG11 routine.  To find out what
| command  number this is, see the table later in this section.
|
|           - BMG function that sends this command to BMG11,
|           | description of arguments sent, and a list of the
|           | possible responses from the 11, if any.
|           |
V           V
```

General initialization and synchronization

Note: All numbers in the Appendix are in octal, not decimal.

```
Restart      BMGReset
               no arguments.
RegDot       BMGRegion
               ((smallNumber regionNumber))
RegTTY       BMGTTYRegion (negative means local printing TTY)
               ((smallNumber regionNumber))
TopTTY       BMGResetTTY
               no arguments.
VidDef       BMGVideoSense
               ((smallNumber planeNumber) (character videoSense (W B)))
TRCOn        BMGTTYCursorEnable
               ((flag))
TRCChar      BMGTTYCursorChar
               ((smallNumber font)(character charCode))
SetBellState BMGBellEnable
               ((smallNumber bellState))
               0 - do nothing
               1 - ring bell
               2 - flash screen
SndAck       (internal to BMG)
               no arguments.
```

Returns a response of no arguments, of type J. Used for synchronization. Note that sending a ↑E (ascii 005) through the TTY stream will also return this response. Thus, to change TTY region parameters the following sequence can be used to ensure that the parameters get changed AFTER all previous TTY output has been displayed, and BEFORE all subsequent TTY output is displayed:

1. Ensure that the TTY stream is empty by sending ↑E through the TTY stream and awaiting the acknowledge.

2. Change the TTY region parameters.

3. Ensure that the paramters are changed by sending a SndAck command through the graphics stream and awaiting the acknowledge.

Region parameter setting commands

REGLIM BMGLocateRegion Does not change current region. The current position for the specified region is reset to 0,0.
((smallNumber regionNumber)
(smallNumber plane)
(number maxX)
(number minX)
(number maxY)
(number minY))

REGDPM BMGDisplayMode
((smallNumber displayMode))
0 = Add (logical or), 1 = remove (clear, i.e. complement and then logical and), 2 = flip (xor), 3 = overwrite (jam), and 4 = inverted overwrite (complement and then jam).

REGFNT BMGFont
((smallNumber font))

REGBKG BMGBackground Does no filling, note.
((word backgroundShade))

REGSCR BMGScrollHeight
((number scrollAmount))
Sets amount that TTY scrolling will scroll by.

RegLSp BMGLineSpacing
((number lineSpacing))
Sets amount of TTY inter-line spacing

RegLMar BMGLeftMargin

```
        ((number leftMargin))
              Sets region-relative left margin for CR
```

Operations that work on all bits in a region

```
REGGRY      BMGFillRegion
              ((word greyPattern))
REGMOV      BMGCopyRegion
              ((smallNumber sourceRegion)
               (smallNumber destinationRegion))
                   Copies bits from one region to another.
                   No reference or change to graphics region.
RegScroll   BMGScrollRegion
              ((number scrollDistance))
                   Scrolls current TTY region. (Note: this does
                   not scroll the current graphics region!)
```

Current position setting, point and line drawing

```
MovAL       BMGMove
              ((number X) (number Y))
MovALX      BMGMove (in just x...)
              ((number X))
MovALY      ...     (in just y)
              ((number Y))
MovRL       BMGMoveRel
              ((number Xincrement) (number Yincrement))
MovRS       BMGMoveRel (by small amount in x and y)
              ((sSmallNumber Xincrement) (sSmallNumber Yincrement))
MovRSX      ...       (by small amount in just x)
              ((sSmallNumber Xincrement))
MovRSY      ...       (by small amount in just y)
              ((sSmallNumber Yincrement))
LinAL       BMGLine
              ((number X) (number Y))
LinALX      BMGLine (in just x)
              ((number X))
LinALY      ...     (in just y)
              ((number Y))
LinRL       BMGLineRel
              ((number Xincrement) (number Yincrement))
LinRS       BMGLineRel (by small amount in x and y)
              ((sSmallNumber Xincrement) (sSmallNumber Yincrement))
LinRSX      ...       (by small amount in just x)
```

```
                ((sSmallNumber Xincrement))
LinRSY          ...       (by small amount in just y)
                ((sSmallNumber Yincrement))
PntAL           BMGPoint
                ((number X) (number Y))
PntALX          BMGPoint (in just x)
                ((number X))
PntALY          ...      (in just y)
                ((number Y))
PntRL           BMGPointRel
                ((number Xincrement) (number Yincrement))
PntRS           BMGPointRel (by small amount in x and y)
                ((sSmallNumber Xincrement) (sSmallNumber Yincrement))
PntRSX          ...       (by small amount in just x)
                ((sSmallNumber Xincrement))
PntRSY          ...       (by small amount in just y)
                ((sSmallNumber Yincrement))
REGSTR          BMGString  ((string))
Circle          BMGCircle ((number radius))
Ellipse         BMGEllipse    For now just a circle with radius = major.
                ((number orientation)
                 (number semiMinorRadius)
                 (number semiMajorRadius))
```

Tablet operations

```
Note that the tablet is always "in" some region, and the tablet
cursor is constrained to be visually within that region.

TbCOn           BMGTabletEnable
                ((flag))
TbCChar         BMGTabletCursorState
                ((character buttonState (U "up", D "down"))
                 (smallNumber tabletFont)
                 (character tabletCharacter))

SetTbReg        BMGTabletRegion ((smallNumber regionNumber))
TbRead          BMGReadTabletPosition     (no arguments)
                  returns a response of type H and arguments:
                    ((character buttonState (U D))
                     (number currentX)
                     (number currentY))
TbEvent         BMGTabletButtonState
                  ((character event (U D))
                   (flag enableOrDisable)
                   (number timeout))    0 means no timeout.
                                        Timeout in 100ths (decimal) of
```

a second.

Font operations

```
InstFont    BMGInstallFont      Font as currently defined will
                                be defined when a BMGReset
                                is done.
              ((smallNumber fontNumber))
              returns a response of no arguments with type:
                E               No room to install the font (intalling
                                implies a copy).
                F               Non-existant fontNumber or an
                                attempt to install TVFONT.
                G               OK -- font is installed.

LoadFont    BMGDefineFont       reserves space for font
              ((smallNumber fontNumber)
               (smallNumber fontHeight)
               (smallNumber fontBaseline))
              returns a response of no arguments with type:
                E               No room for a new loaded font.
                F               Font cannot be loaded: either no such
                                font number or is already loaded.
                G               OK -- font is loaded.

FreeFont    BMGFreeFontNumber   ((smallNumber fontNumber))
              returns response of no arguments with type:
                F                 No such font number or undefined.
                G                 OK -- font is freed.

LoadChar    BMGDefineFontChar
              Note that the character width may be at most 16 decimal.
              ((smallNumber NumberOfBytesToFollow)
               (smallNumber fontNumber)
               (smallNumber charCode)
               (smallNumber leftKern)
               (smallNumber charWidth)
               (word rasterLine)..) {as many as in height(font)}
              returns a response of no arguments and type:
                D               Number of raster lines coming does not
                                match the font's height.
                E               No room to load another character.
                F               No such font number or undefined.
                G               OK -- character is defined.
```

General input operations

```
(internal BMG function)
  ((smallNumber NumberOfBytesToFollow)
   (string preDelim)
   (string postDelim))
        [Note that number of bytes includes the two
         string terminators.  Also note that these
         delimiter strings are not allowed to contain
         nulls, ascii 0.  They may contain octal 200
         though, which may amount to the same effect
         as far as the host is concerned.]
        Any message from the /11 to the /10 is
        preceeded by the preDelim and followed by the
        postDelim.  Initial values are:
                preDelim  = "↑A("
                postDelim = ")<CRLF>"
BMGSetTimer  ((number timeOutLimit))
```

VT52 emulation

```
aVT52       Enter vt52 simulation. (not used by BMG)
              no arguments.
```

Display Macros and Display Processes

```
DMPause     BMGPause
              Pause the current process for a certain time,
              specified in 100ths (decimal) of a second.
            ((number time))

DMDefine    BMGDefineMacro
              Start a definition of a display macro.  All
              bytes sent after the first argument, the
              DMID, become part of the display macro,
              up to the terminator -- an 026 (decimal)
              byte followed by an 027 (actually any
              non-026). The two-byte sequence 026 026
              turns into a single 026 that will be part
              of the display macro.  (i.e.  026 026 is
              a "quoted 026".)
            ((smallNumber DMID)
```

```
                (byte byteOfMacro)...
                026 027)

              returns a response of type:
                G           OK, display macro defined.
                E           No room, display macro not defined.
                            e.g.  couldn't allocate needed space
                            for the display macro bytes, or for
                            the internal definition block.
```

DMExecute BMGExpandMacro

```
                Execute a display macro n times at a given frequency.
                (The interval from the start of one execution of the
                display macro to the start of the next is specified,
                in 100ths of a second.)
              ((smallNumber DMID)
               (smallNumber iterationCount)
               (number iterationInterval))
```

DMTbExecute (internal BMG function)

```
                Execute one of two display macros depending
                on the state of the table (button-up or
                button-down).  The current position is first
                moved to be at the current tablet position
                before executing the display macro.
                  ((smallNumber DMIDup)
                   (smallNumber DMIDdown)
                   (smallNumber iterationCount)
                   (number iterationInterval)
```

DMKill BMGKillMacros

```
                Kill all display macro definitions.  (The
                implementation had to take some shortcuts
                to finish in time and this is one: no garbage
                collector, no freeing of just part of the
                display macro storage.)
```

DPCreate BMGSpawnProcess

```
                Create and start a display process, telling it to
                execute a given display macro n times at a given
                frequency (as if the display process had a top level
                DMExecute command) and use a given graphics region
                initially.  When the display process is done, it will
                kill itself.  If the specified iteration count is 0,
                the display process will execute the display macro
                indefinitely.  (Until killed by a DPKill, or a
                BMGReset is done, etc.)
              ((smallNumber DPID)     {the new display process's ID}
               (smallNumber region)
```

```
        (smallNumber DMID)      {what display macro to execute}
        (smallNumber iterationCount)
        (number iterationInterval))

      returns a response of type:

        G           OK, process created and running.
        E           No room, process not created.  i.e., no
                    room to allocate the internal overhead
                    (process control block, display
                    process control block) or if the DPID
                    is already in use by some other
                    display process.

DPKill      BMGKillProcess
              Kill a running display process, given its ID.
            ((smallNumber DPID))
```

A.2 Character Sequence Syntax

The following describes the character sequences sent over the graphics stream for various command formats. Note that the characters sent over the graphics stream are not necessarily the exact characters sent to the PDP11. For instance, some characters must be quoted to get into the graphics stream, e.g. to get a ↑R (022 octal) into the graphics stream, the two character sequence ↑R R (022 122) must be sent. These escape sequences for quoting (and others for stream-switching) are thus not considered part of the graphics stream -- they are at a lower level than streams.

Argument description legend

<argument-list> :: = "(" <argument> <argument> ... ")"

<argument> :: = "(" <type> <optional-name> <optional-value-list> ")"
Optional-value-list gives the possible
values that may be used for this argument.

<type> :: = character | flag | string | smallNumber |
sSmallNumber | number | word

Type Formats

character one 8-bit byte in the range 0 through and including 200 octal. 8-bit values above that are considered the same as if they had their 200-bits off, so except for the 200 value, characters are only 7 bits.

flag one character, either "Y" or "N".

↑A(A xCoordinate yCoordinate)
Downward button transition.

↑A(B xCoordinate yCoordinate)
Upward button transition.

↑A(C buttonState xCoordinate yCoordinate)
Timeout waiting for button transition.

↑A(D) Error from LoadChar: number of raster lines coming in character definition does not match height of font.

↑A(E) No room for requested operation. i.e. BMG11 cannot allocate space for a new font, character, or process. (Note that installing a font implies copying it.) Sent by InstFont, LoadFont, and LoadChar.

↑A(F) No such value allowed. Either an argument was out of range (i.e. fonts are 0-16) or the value is in a legal range but specifies something not defined, e.g. trying to load a character into an unloaded font. Or, attempting to install font 0, TVFONT, which is pre-installed and not allowed to be altered. Sent by InstFont, LoadFont, and LoadChar.

↑A(G) Acknowledgment -- the operation was successful. Sent by InstFont, LoadFont, and LoadChar.

↑A(H buttonState xCoordinate yCoordinate)
Response to command from host to read the tablet. This is a self-consistent state for some time after the read-tablet command and before the time sent by BMG11. When received and processed by BMG routines (and their callers), the current tablet state may be different, so care is necessary in using this.

↑A(I) The timer (set by BMGSetTimer) went off. (It goes off at most once for each time it is set. If another BMGSetTimer resets it, the old setting is lost.)

↑A(J) A response to a TTY or graphics stream acknowledge request. Can be used for synchronizing operations between streams, e.g. changing TTY region parameters.

A.3 Low-level TTY/Graphics stream switching

All input (including nulls (0) and rubouts (177), which are fine characters) is buffered in one of two buffers: TTY input buffer, TIB, or graphics input buffer, GIB. We simulate what we would really like, which is two communication lines or at least two decent end-to-end protocol-controlled streams. Since we don't have the requisite stream support on the host side, in Interlisp/10, we have a simple approximation to a good protocol: there are some reserved character sequences which switch between the two streams, handled at interrupt-time:

↑RT (Control-R followed by T) will switch to the TTY stream.
↑RG will switch to the graphics stream.
↑RU (U for "up") PUSHes to the TTY stream, explained below.
↑RD (D for "down") POPs from the TTY stream, explained below.

(Control-R was chosen so not to be a small and frequent number -- like Control-A, which is what we had before. It should also have an even parity so its 200-bit won't be turned on by the simple printing that might do TTY pushes.)

Consecutive ↑Rs act as one ↑R -- in other words, if x is any character, then ↑R...↑Rx does whatever ↑Rx does. This is to add robustness in the face of an outstanding ↑R followed by the death or interruption of the program that sent it.

The interrupt-level routines must also be able to manage the ↑S/↑Q "flow-control" protocol, in two directions if enabled: the host operating system (TOPS20, presumably) can be told that we will send a ↑S when (either of) our input buffers are nearly full and ↑Q when they are (both) nearly empty. The ↑S tells the host to stop sending until it gets a ↑Q. That can be turned off, which is lucky since this removes ↑S and ↑Q from the set of useful characters you can send through to some application program on the host -- e.g. EMACS (and our input buffers -- the TTY stream one is all that matters for EMACS -- are large enough so the flow-control isn't needed). In the other direction, things are not so flexible: the host will send us a ↑S when its (small -- e.g. around 40 characters) input buffer is nearly full, and that does not seem to be able to be turned off. Thus we must lose ↑S and ↑Q from the set of characters that the host can send us.

Note that the host sends us ↑S with the "parity" bit (i.e. 200 bit) ON (for TOPS20 Release 4 and later) and OFF for TENEX and TOPS20 before Release 4. We have the variable Ctl.S to specify which -- right now the only way to switch between them is in DDT e.g. to set it type "ctl.s/023" followed by a return.

Since we thus lose ↑Q (021) and ↑S (023 on some systems, 223 on others) (in addition to ↑R, our escape command) we need some low-level quoting to allow ↑R, ↑Q, and ↑S to go through a stream. In these quote sequences, the second character is formed by logically ORing in 100 octal, with the 200 bit being ignored:

```
↑RR (Control-R R, or 022 122, or 022 322) quotes a ↑R, octal 022 --
    i.e.  it turns into a single Control-R in whichever is the
    currently selected stream.
↑RQ (Control-R Q, or 022 121 etc.) quotes a ↑Q, octal 021.
↑RS (Control-R S, or 022 123 etc.) quotes a ↑S, octal 023 or 223,
    whichever Ctl.S is set to.
```

Octal 222, 221, and either 023 or 223 (whichever Ctl.S is NOT set to) are like normal characters and need no special action.

Pushing to and popping from the TTY stream is designed for stupid (i.e. minimally controlled), unexecpected output, for instance the low-level Interlisp messages generated by a Control-T or garbage collection. Pushing to the TTY stream switches to the TTY stream, pushing the current stream. Popping restores the last stream pushed. Since there is no state (essentially -- except for TTY font-switching, but that's very safe) in the TTY stream, pushing from either stream is simply a matter of remembering which of the two streams it was. There is no way to push to the graphics stream, as that stream has a large amount of state. Thus, stupid output cannot do anything fancy -- i.e. it cannot use the graphics stream.

Switching between streams via ↑RT and ↑RG resets the stream-stack, in case the stupid ouputters are losing control. The stream-stack is simply an up-down counter, TPushCount, of how many times we have pushed to the TTY stream, and a variable, TGBase, that says whether we started pushing from the TTY or graphics stream. The currently selected stream is recorded by the PC of the input interrupt routine's coroutine.

A.4 Assignment of command numbers

The following is the dispatch table for BMG commands. Commands are indexed by an 8-bit byte, so there are 256 commands -- though a lot of them are illegal in that they go to BadCom and it prints a complaint and then tries to resynchronize by ignoring all characters in the graphics stream until that stream is empty -- assuming that that point is likely to be a graphics com start. The reason it has to do this is that if the command is unknown, its syntax is too -- and we don't know how many of the following bytes are arguments to that unknown command and how many are succeeding commands.

Restart is command 0, TopTTY is command 1, etc.

```
DspchT:
    NoOp                ; NoOp
    TopTTY              ; BMGResetTTY
    REGDOT              ; BMGRegion
    REGTTY              ; BMGTTYRegion
    VIDDEF              ; BMGVideoSense
    SetMsgDelims        ; BMGSetInputDelimiters
    SetTimer            ; BMGSetTimer
    SndAck              ; end BMGGraphicsAcknowledge (↑E in TTY stream)
    Restart             ; BMGReset
    REGLIM              ; BMGLocateRegion
    REGDPM              ; BMGDisplayMode
    REGFNT              ; BMGFont
    REGBKG              ; BMGBackground
    RegSHeight          ; BMGScrollHeight
    RegLSp              ; BMGLineSpacing
    RegLMar             ; BMGLeftMargin
    REGGRY              ; BMGFillRegion
    REGMOV              ; BMGCopyRegion
```

```
RegScroll       ; BMGScrollRegion
RegChar         ; BMGChar
REGSTR          ; BMGString
MovAL           ; BMGMove
MovALX          ; BMGMove (in just x...)
MovALY          ; ...     (in just y)
MovRL           ; BMGMoveRel
MovRS           ; BMGMoveRel (by small amount in x and y)
MovRSX          ; ...        (by small amount in just x)
MovRSY          ; ...        (by small amount in just y)
LinAL           ; BMGLine
LinALX          ; BMGLine (in just x)
LinALY          ; ...     (in just y)
LinRL           ; BMGLineRel
LinRS           ; BMGLineRel (by small amount in x and y)
LinRSX          ; ...        (by small amount in just x)
LinRSY          ; ...        (by small amount in just y)
PntAL           ; BMGPoint
PntALX          ; BMGPoint (in just x)
PntALY          ; ...      (in just y)
PntRL           ; BMGPointRel
PntRS           ; BMGPointRel (by small amount in x and y)
PntRSX          ; ...         (by small amount in just x)
PntRSY          ; ...         (by small amount in just y)
Circle          ; BMGCircle
Ellipse         ; BMGEllipse
InstFont        ; BMGInstallFont
LoadFont        ; BMGDefineFont
FreeFont        ; BMGFreeFontNumber
LoadChar        ; BMGDefineFontChar
TRCOn           ; BMGTTYCursorEnable
TRCChar         ; BMGTTYCursorChar
TbCOn           ; BMGTabletEnable
TbEvent         ; BMGTabletButtonState
TbCChar         ; BMGTabletCursorState
SetTbReg        ; BMGTabletRegion
TbRead          ; BMGReadTabletPosition
aVT52           ; Enter vt52 simulation.
SetBellState    ; BMGBellEnable.
DMPause         ; BMGPause
DMDefine        ; BMGDefineMacro
DMExecute       ; BMGExpandMacro
DMKill          ; BMGKillMacros
DPCreate        ; BMGSpawnProcess
DPKill          ; BMGKillProcess
```

REFERENCES

Sproull, R. F. Raster Graphics for Interactive Programming Environments. *Computer Graphics*, August 1979, 13(2), 83-93.

Teitelman, W. *A Display Oriented Programmer's Assistant*, pages 905-915. IJCAI, Cambridge, MA, 1977.

Teitelman, W. *INTERLISP Reference Manual*. Revised October 1978 edition, Xerox Palo Alto Research Center, Palo Alto, CA, 1978.

INDEX